**For each of these projects, carry out the following steps:**

1. **Work out an overall design strategy for the system and draw block diagrams.**

1. **Divide the system into modules if appropriate. Develop an algorithm, SM charts, or state graphs as appropriate for each module. Unless otherwise specified, your design should be a synchronous system with appropriate circuits added to synchronize the inputs with the clock.**

1. **Write synthesizable VERILOG code for each module, simulate it, and debug it. Use test benches to verify correct operation of each module.**

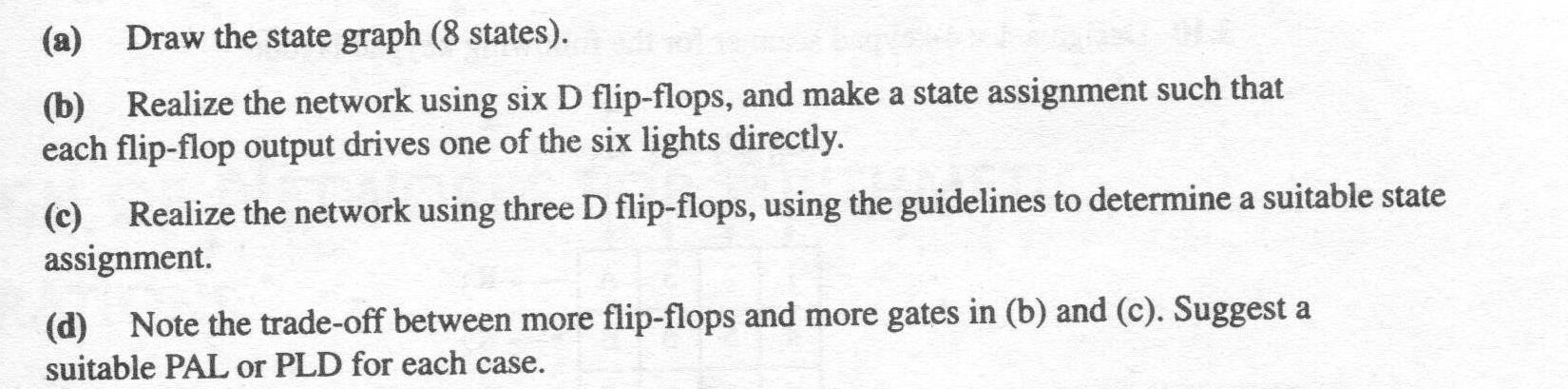
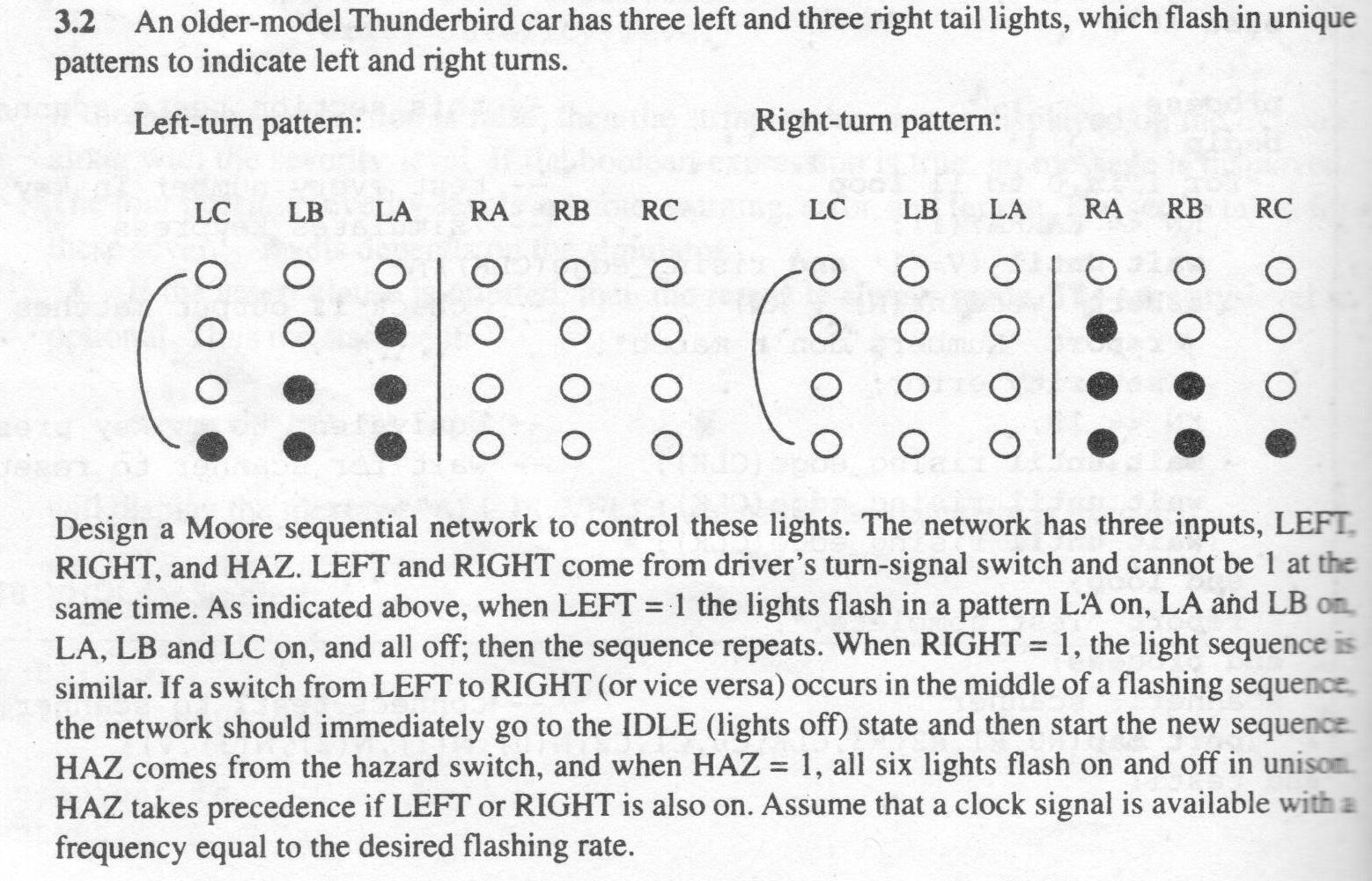
1. **Design has multiple solutions and has innovation component. Only extraordinary, innovative designer will get highest grade. 5% of your grade will be based on creativity/uniqueness you add to your design.**

1. **Mini projects under Digital System Design EE311 (Non lab component design work out) shall be submitted as an assignment. [Evaluation for a total of 15 marks – should include block diagram, detailed flow diagram, State graphs, controller architecture and design].**

1. **Mini projects under Digital System Design EE335 (Lab component design work out) shall be submitted as a project folder with your roll numbers.[Evaluation for a total of 40 marks – should include separate files on Verilog HDL code for all the modules used, timing analysis, test files and a small documentation].**

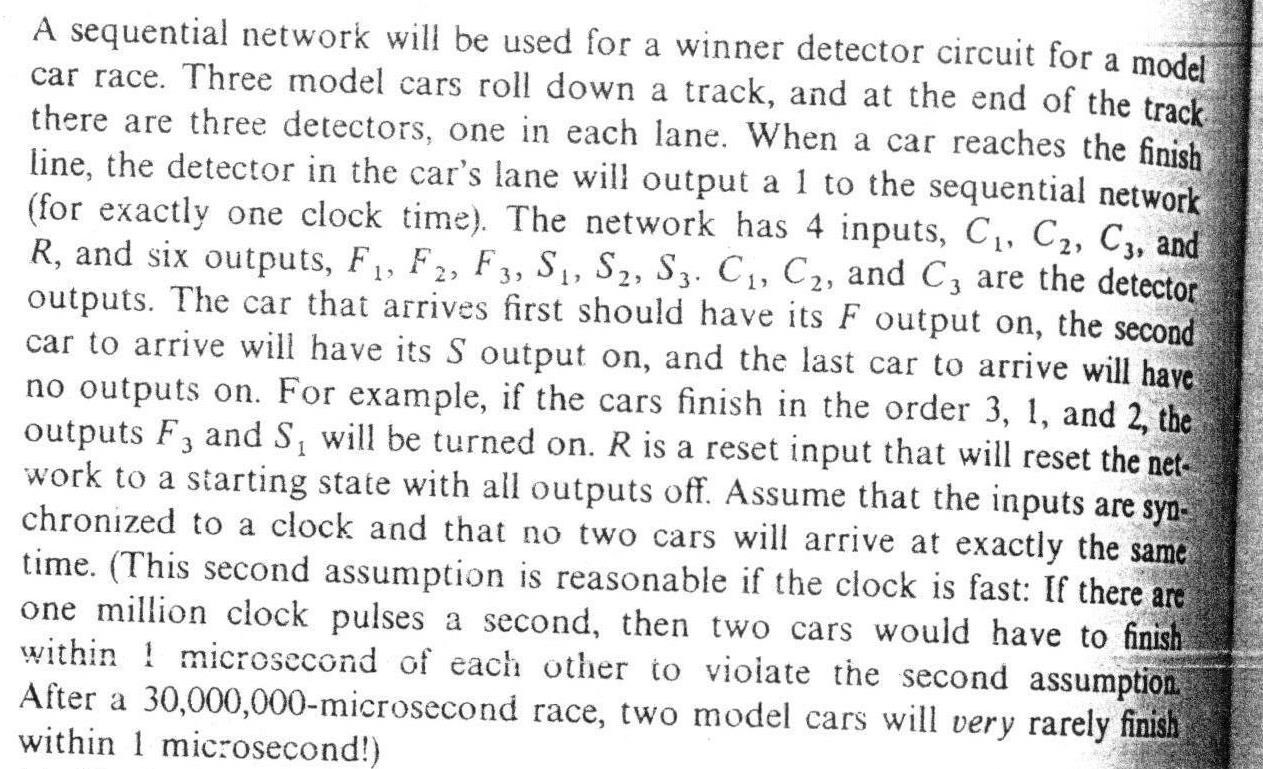
|  |  |  |  |
| --- | --- | --- | --- |
| Sl. No. | Register No. | Name of the Student | Mini project no. |
| 1 | 16709816EE129 | Adarsh Malapaka | 1 |
| 2 | 16618416EE201 | Adithya M | 2 |
| 3 | 16615016EE202 | Aditya R | 3 |
| 4 | 16652416EE103 | Aditya Rangarajan | 4 |
| 5 | 16641516EE125 | Akshata Ashok Kini | 5 |
| 6 | 16624116EE153 | Avinash Kumar | 6 |
| 7 | 16645416EE154 | Chitransh Lodha | 7 |
| 8 | 16623316EE218 | Gayatri Indukumar | 8 |
| 9 | 16710516EE222 | Josh Pius | 9 |
| 10 | 16659416EE126 | Kodipyaka Saimeghana | 10 |
| 11 | 16643816EE225 | Kshitiz Khatri | 11 |
| 12 | 16657016EE127 | Lekshmi Sadanandan | 12 |
| 13 | 16663116EE128 | Mahima Verginia Rebello | 13 |
| 14 | 16620716EE130 | Manish S Babu | 1 |
| 15 | 16632816EE120 | Mansi joisher | 2 |
| 16 | 16610016EE234 | Megh Manoj Bhalerao | 3 |
| 17 | 16643916EE134 | Munna Kumar | 4 |
| 18 | 16644816EE235 | Nandani Shakshi | 5 |
| 19 | 16646316EE230 | Naveen Kumar | 6 |
| 20 | 16632216EE236 | Neelam Bugalia | 7 |
| 21 | 16665916EE155 | P. Uday Kumar Reddy | 8 |
| 22 | 16653116EE237 | Pacharla Sreedhar Reddy | 9 |
| 23 | 16642216EE124 | Rohit Satish Karmarkar | 10 |
| 24 | 16631016EE139 | S.Manish | 11 |
| 25 | 16603516EE140 | Saraswathi Vinod | 12 |
| 26 | 16636716EE253 | Satyajit Gantayat | 13 |
| 27 | 16621516EE254 | Saurabh Shankar Zond | 1 |
| 28 | 16601216EE244 | Shama G Vasisht | 2 |
| 29 | 16652716EE245 | Sharanya B | 3 |
| 30 | 16625816EE142 | Shruti Deshpande | 4 |
| 31 | 16663516EE144 | Sneha Jogdhankar | 5 |
| 32 | 16647716EE146 | Sri Vishnu S | 6 |
| 33 | 16631716EE147 | Sujith D Dixith | 7 |
| 34 | 16638816EE149 | Vaishakha | 8 |
| 35 | 16706816EE151 | Vinayak Anil Kumar | 9 |
| 36 | 16608616EE209 | Vrushabh Bhangod | 10 |
| 37 | 16640216EE251 | Yandrapu Rahul Raja | 11 |

1.



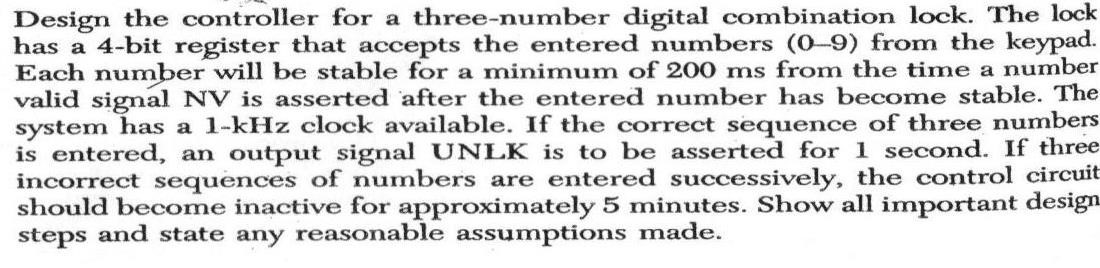
(e) Write a Verilog code and simulate your solution to (b).

2.



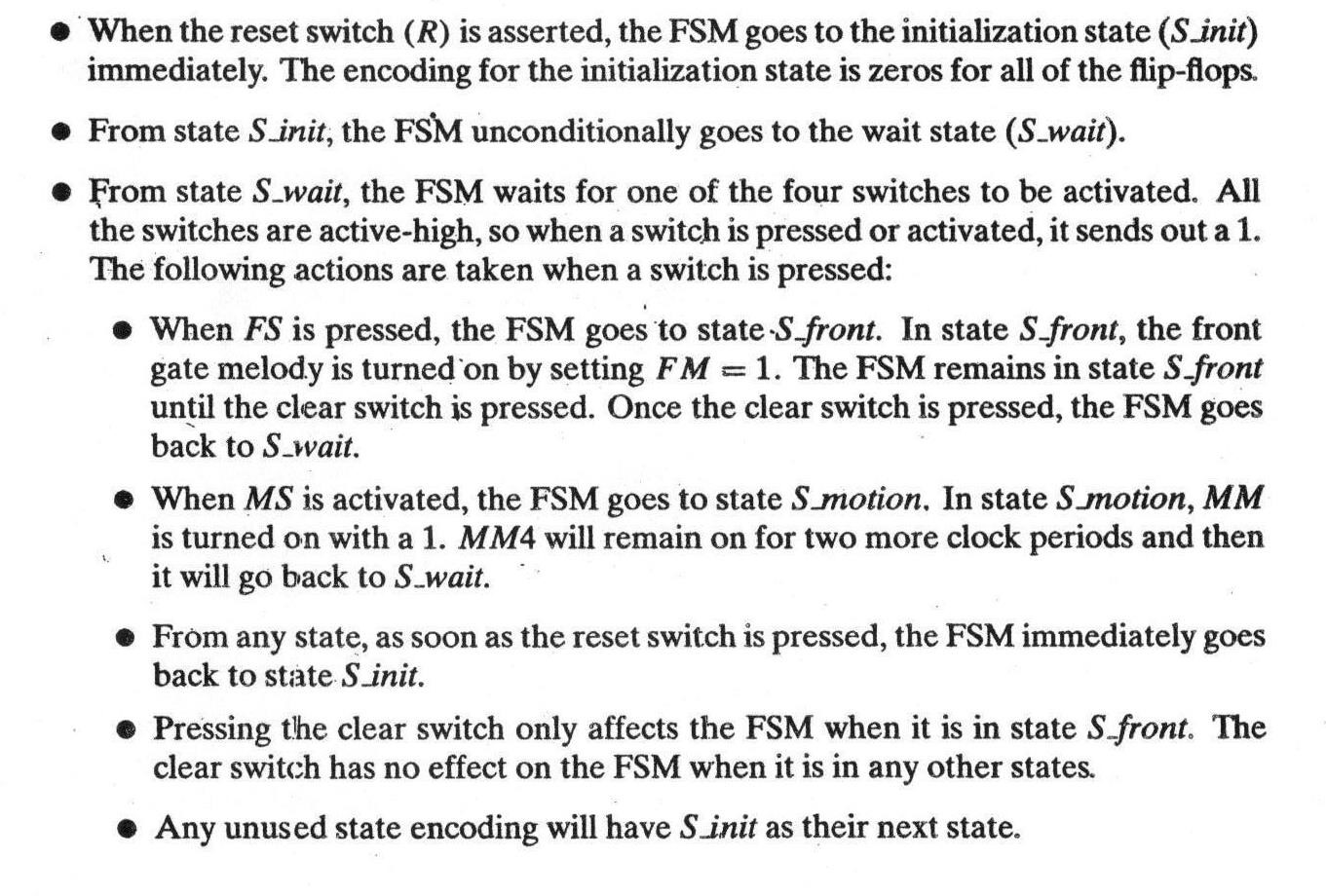
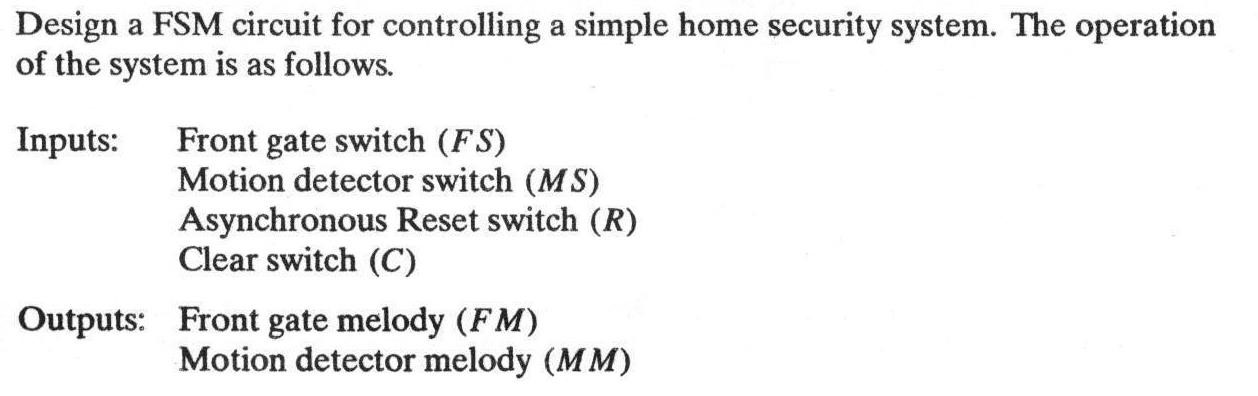
1. Find a Moore state graph for the network (10 states).
2. Write a Verilog code and simulate your solution to (a).

3.



Write a Verilog code and simulate your solution.

4.



Write a Verilog code and simulate your solution.

* 1. **Push-Button Door Lock**

Design a push-button door lock that uses a standard telephone keypad as input. Use the keypad scanner as a module. The length of the combination is 4 to 7 digits. To unlock the door, enter the combination followed by the # key. As long as # is held down, the door will remain unlocked and can be opened. When # is released, the door is relocked. To change the combination, first enter the correct combination followed by the \* key. The lock is then in the “store” mode. The “store” indicator light comes on and remains on until the combination has been successfully changed. Next enter the new combination (4 to 7 digits) followed by #. Then enter the new combination a second time followed by #. If the second time does not match the first time, the new combination must be entered two times again. Store the combination in an array of eight 4-bit registers or in a small RAM. Store the 4-bit key codes followed by the code for the # key. Also provide a reset button that is not part of the keypad. When the reset button is pushed, the system enters the “store” state and a new combination may be entered. Use a separate counter for counting the inputs as they come in. A four-bit code, a key-down signal (*Kd*), and a valid data signal (*V*) are available from the keypad module.

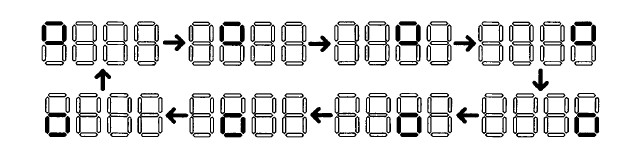
Write a Verilog code and simulate your solution.

* 1. Design a controller that switches on lights, air conditioner(s), and other power points in a mini-concert hall as the first person enters the hall and switches off the lights and other devices as the last person leaves it. The concert hall has four entry/exit doors equipped with infra-red sensors. Only one person can enter or leave through a door at a time. Realize Verilog design and test it.

* 1. A driver less shuttle, a light rail car, which plies between two airports at a distance of 2 miles, is to be controlled automatically. When it is waiting for the passengers at one of the stations, the two entrance/exit doors of the car must remain open. So also, the corresponding doors at the station. After the car comes to a halt at a station, the car doors as well as the station doors open. In each of the stations, a push button is installed for use by the passenger(s) to request service of the car which is waiting for passengers at the other station, and has radio linked switches to detect the requests. The car leaves a station after 10 min of arrival, provided there is at least one passenger in the car at the time of departure. At the appointed time of departure, if there is no passenger in the car and, if a service request from the other station is pending, all the doors of the car and the station close, and the car departs to the other station without passengers. However, if there is no request pending, the car waits for the passengers with doors shut. When a passenger arrives, the passenger is allowed to get in and the car departs. At the time of closing, if any passenger arrives, the doors open for 5 s and close again, provided the car is not full. The entry to the car or exit from the car can be made through any of the two doors. Each of the two doors allows only one person at a time. The car can carry a maximum of 25 passengers. Draw a detailed specification of the controller and design the architecture so that the design may be coded in Verilog. State your assumptions clearly.

* 1. In a seven-segment LED display, a square pattern can be created by enabling the a, b, f, and g segments or the c, d, e, and g segments. We want to design a circuit that circulates the square patterns on the seven-segment LED displays. The clockwise circulating pattern is shown in Figure below. The circuit should have an input, en, which enables or pauses the circulation, and an input, cw, which specifies the direction (i.e., clockwise or counterclockwise) of the circulation.

Design the circuit and code using Verilog and verify its operation. Make sure that the circulation rate is slow enough for visual inspection.



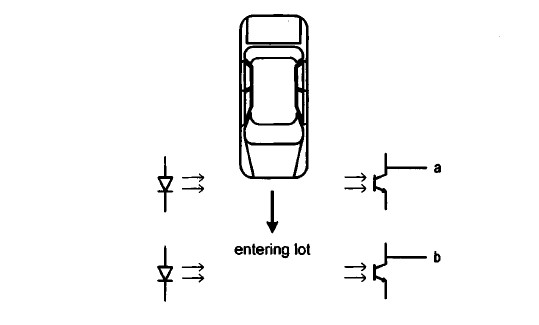
* 1. A programmable square-wave generator is a circuit that can generate a square wave with variable on (i.e., logic 1) and off (i.e., logic 0) intervals. The durations of the intervals are specified by two 4-bit control signals, m and n, which are interpreted as unsigned integers. The on and off intervals are m\*100 ns and n\*100 ns, respectively. Design a programmable square-wave generator circuit. The circuit should be completely synchronous. Write the corresponding Verilog code and test using a logic analyser to verify its operation.

* 1. Consider a parking lot with a single entry and exit gate. Two pairs of photo sensors are used to monitor the activity of cars, as shown in Figure below. When an object is between the photo transmitter and the photo receiver, the light is blocked, and the corresponding output is asserted to 1. By monitoring the events of two sensors, we can determine whether a car is entering or exiting, or a pedestrian is passing through. For example, the following sequence indicates that a car enters the lot:

* + - Initially, both sensors are unblocked (i.e., the a and b signals are "00").
    - Sensor a is blocked (i.e., the a and b signals are "10"). • Sensor a is unblocked (i.e., the a and b signals are "01").
    - Both sensors become unblocked (i.e., the a and b signals are "00").

Design a parking lot occupancy counter as follows:

* + 1. Design an FSM with two input signals, a and b, and two output signals, **enter** and **exit.** The **enter** and **exit** signals assert one clock cycle when a car enters and one clock cycle when a car exits the lot, respectively.
    2. Derive the HDL code for the FSM.
    3. Design a counter with two control signals, **inc** and **dec,** which increment and decrement the counter when asserted. Derive the HDL code.
    4. Combine the counter and the FSM and seven-segment LED decoding circuits. Use two push buttons to mimic operation of the two sensor outputs. Verify operation of the occupancy counter.



* 1. The rising-edge detector is a circuit that generates a short one-clock-cycle tick when the input signal changes from 0 to 1. It is usually used to indicate the onset of a slow time varying input signal. A dual-edge detector is like a rising-edge detector except that

the output is asserted for one clock cycle when the input changes from 0 to 1 (i.e., rising edge) and 1 to 0 (i.e., falling edge).

* + 1. Design a circuit based on the Moore machine and draw the state diagram and ASM chart.
    2. Derive the HDL code based on the state diagram of the ASM chart.
    3. Derive a test bench and use simulation to verify operation of the code.
    4. Repeat steps 1 to 3 for a Mealy machine-based design.

* 1. A train station has three platforms marked 1 to 3. A train approaching the station in any of the two directions is to be routed to one of the three platforms. If all the three platforms are empty, then the approaching train is to be routed to platform 1. On the other hand, if it is occupied, the train is to be routed to platform 2. Only if both platforms 1 and 2 are occupied, the train is routed to platform 3.

A switching control system is required to be designed which will set the appropriate rail track points and turn on signal lights. Each platform has a sensor which is turned on if a train is in that platform. If a train approaches a light signal, the corresponding sensor is activated. The controls required for departing trains from the platforms may be ignored. What is the easiest way to design? Design the circuit and code using Verilog and verify its operation.

* 1. Construct a digital clock using counters. The clock displays hour and minute digits. The user can adjust the time by buttons. Write the Verilog description of the digital clock module. This module has five inputs. These are clk (main clock signal), en (active high enable signal), rst (resets all outputs when in logic level 1), hrup and minup (adjust hour and minute values). The digital clock module has six outputs each with four bits. These are s1 and s2 (for second digits), m1 and m2 (for minute digits), h1 and h2 (for hour digits).